What is claimed:

1. A method of transport protocol optimization of an internet protocol, comprising the steps of (a) using a source packet interceptor to intercept an IP packet from a source application, (b) using a source edge process to act as the new destination for the source application, (c) using a source packet driver to aggregate the intercepted IP packets, (d) using a source data mover to transport the aggregated IP packets over a communication link to a destination data mover, (e) using a destination packet driver to disaggregate the transported aggregated packets, (f) using a destination edge process to deliver the disaggregated IP packets to a destination application.

2. The transport protocol optimization method of claim 1, comprising the step of using IP routing.

3. The transport protocol optimization method of claim 2, wherein the IP packet is optionally a TCP, UDP, ICMP, or other type of IP packets.

4. The transport protocol optimization method of claim 3, wherein intercepting an IP packet from the source application comprises the steps of comparing the IP packet's address to packet addresses in a look-up table and (b) intercepting only those source packets with the same addresses as those stored in the look-up table.

5. The transport protocol optimization method of claim 4, wherein the address of the IP packet comprises the packet's source IP address, source port number, destination IP address, destination port number, and protocol type.

6. The transport protocol optimization method of claim 5, wherein intercepting an IP packet from the source application comprises the step of routing the IP packet to an edge process that is exclusive to the address of the IP packet.

7. The transport protocol optimization method of claim 6, wherein intercepting an IP packet from the source application comprises the steps of an edge process (a) reading the data contained in the routed IP packets and (b) forming a message header field for the routed IP packets.

8. The transport protocol optimization method of claim 7, comprising the step of the packet driver forming a packet driver message.

9. The transport protocol optimization method of claim 8, wherein the packet driver message comprises the message header field and intercepted IP packet data from one edge process.

10. The transport protocol optimization method of claim 9, comprising the step of forming a plurality of packet driver messages.

11. The transport protocol optimization method of claim 12, comprising the step of aggregating multiple packet driver messages into a packet driver buffer.

12. The transport protocol optimization method of claim 11, wherein the size of the aggregated packet driver messages is less than or equal to a predetermined maximum size of the buffer.

13. The transport protocol optimization method of claim 12, comprising the step of the packet driver forming a routing header in the packet driver buffer that precedes the first packet driver message.

14. The transport protocol optimization method of claim 13, wherein the routing header comprises a function type field, a number of packet driver messages field, and a data length field.

15. The transport protocol optimization method of claim 14, wherein the message header comprises a version field, a length of header field, a message function type field, a message flag field, a protocol type field, a sequence number field, a source IP address field, a destination IP address field, a source IP port number field, a destination IP port number field, a length of data field, and a status field.

16. The transport protocol optimization method of claim 15, comprising the step of combining the routing header field, the message header field, and the intercepted IP packet data from one edge process.

17. The transport protocol optimization method of claim 16, comprising the step of using a compression engine to compress the packet driver buffer.

18. The transport protocol optimization method of claim 17, comprising the step of routing the packet driver buffer to the data mover.

19. The transport protocol optimization method of claim 18, wherein transmission of packet driver buffers over a communication link by the data mover comprises the step of (a) inserting data mover protocol fields into the start of the packet driver buffer; (b) if necessary, reducing the size of the packet driver buffer by breaking the buffer into multiple segments, with each segment being no greater than the size specified in the configuration file; (c) using standard UDP socket calls to interface with the TCP stack for UDP delivery of the segments over the network.

20. The transport protocol optimization method of claim 19, wherein the communication link is comprised of a UDP link.

21. The transport protocol optimization method of claim 20, wherein the data mover protocol comprises (a) data mover transport data subfield, and (b) data mover transport acknowledgement subfield.

22. The transport protocol optimization method of claim 21, wherein the data mover transport data subfield comprises the length of the entire subfield, the subfield type code, the logical sequence number of this transport message, and the physical sequence number of this transport message.

23. The transport protocol optimization method of claim 22, wherein the data mover transport acknowledgement subfield comprises the length of the entire subfield, the subfield type code, the highest physical block number sent from this side of the connection, the highest physical block number received on this side of the connection, the bit-significant flags representing the blocks received, and the rate of data delivery to the remote packet driver.

24. The transport protocol optimization method of claim 23 wherein packets are intercepted by the Linux exit point: NF_IP_PRE_ROUTING.

25. The transport protocol optimization method of claim 24 comprising the step of modifying the destination address of the IP packets accepted for interception to be the address of the source packet interceptor.

26. The transport protocol optimization method of claim 25, comprising the step of creating a edge process for each TCP application connection; a UDP edge process for each UDP intercept; and a ICMP edge process for a ICMP intercept.

27. The transport protocol optimization method of claim 26, comprising the step of terminating any connection between a source application and a destination application.

28. The transport protocol optimization method of claim 27, comprising the step of opening a connection between a source data mover and a destination data mover.

29. The transport protocol optimization method of claim 28, comprising the steps of (a) opening a connection between the source application and the source edge processor and (b) opening a connection between the destination edge processor and the destination application.

30. The transport protocol optimization method of claim 29, wherein the UDP connection for transporting the stored packets is over a WAN.

31. The transport protocol optimization method of claim 30, comprising the steps of (a) transporting packets from the source application to the source packet interceptor

over a source LAN and (b) transporting packets delivered to a destination data mover to a destination application over a destination LAN.

32. The transport protocol optimization method of claim 31, wherein the decompression engine also performs the step of enabling or disabling compression as a function of feedback of the instantaneous level of compression.

33. The transport protocol optimization method of claim 32, wherein optimization is comprised of the step of optimization using transport protocol optimization source software and destination software.

34. The transport protocol optimization method of claim 33, wherein, the source software optionally runs on a source server, a source network switch, or as a source network appliance and the destination software optionally runs on a destination server, a destination network switch, or as a destination network appliance.

35. The transport protocol optimization method of claim 34, comprising the step of optionally connecting the source and destination network appliances to a (a) network switch, which switch is connected to an application server running a application; (b) network switch, which switch is connected to an application server running a application and to a network router; or (c) to an application server running a application.

36. The transport protocol optimization method of claim 34, comprising the step of integrating the source packet interceptor, driver, end processors, compression engine, and data mover into a source TPO.

37 The transport protocol optimization method of claim 34, comprising the step of integrating the packet interceptor, driver, end processors, compression engine, and data mover into a destination TPO.

38. The transport protocol optimization method of claim 37, comprising the step of using a source TPO and a destination TPO to create a pair of TPOs.

39. The transport protocol optimization method of claim 38, comprising a plurality of pairs of TPOs optionally for multicasting and for multipoint communication.

40. The transport protocol optimization method of claim 39, comprising the steps of (a) attaching a source server running the source application on a source LAN, (b) attaching a source TPO on the source LAN and, (c) attaching a destination server running a destination application on a destination LAN, and (d) attaching a destination TPO on the destination LAN.

41. The transport protocol optimization method of claim 40, wherein the packets from the source application are transported over the source LAN to the source TPO and

the packets from the destination application are transported over the destination LAN to the destination TPO.